MetaCase

Using Domain-Specific Modeling for Design and Verification of Cyber Physical Systems

17 March, 2016

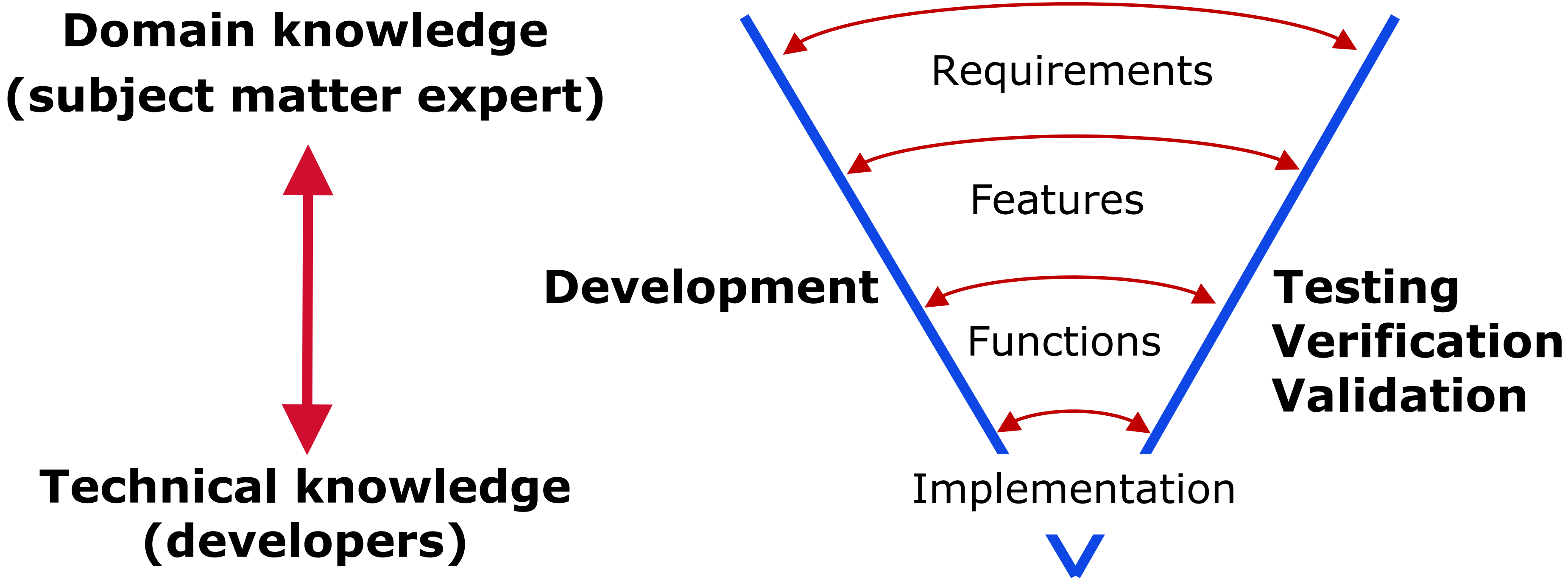
Juha-Pekka Tolvanen
jpt@metacase.com

Motivation

- Domain expert vs. verification expert
  - Domain experts are not necessarily familiar with development and testing tools
  - Developers and verification experts don’t always master the domain knowledge

- Domain-specific languages enable to use directly the domain concepts for both development and V&V
  - Common vocabulary to enable feedback and communication
  - Higher level languages improve productivity
  - Automated transformations improve quality

Separate vs. common language



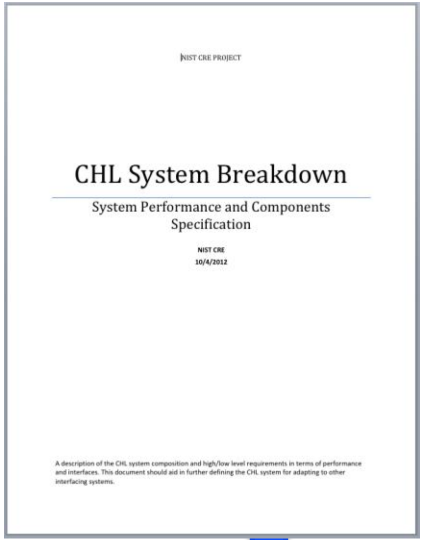
Domain-Specific Modeling (DSM)

- Use of concepts from the problem domain
  - Already familiar => no need to learn new
  - Have known semantics
- Having a special focus
  - Use concepts that are relevant for the task: development, testing, verification, validation
- Use concrete syntax that enables communication and collaboration close to domain's natural representation
  - Not a cryptic programming/scripting language
- DSM is applied in particular for automating repetitive development efforts*, but less in testing and V&V

* See references on EADS, NSN, Nokia, Panasonic, Polar, Elektrobit, USAF

Example: Industrial Process Plant

Example Specification



NIST CRE PROJECT

CHL System Breakdown

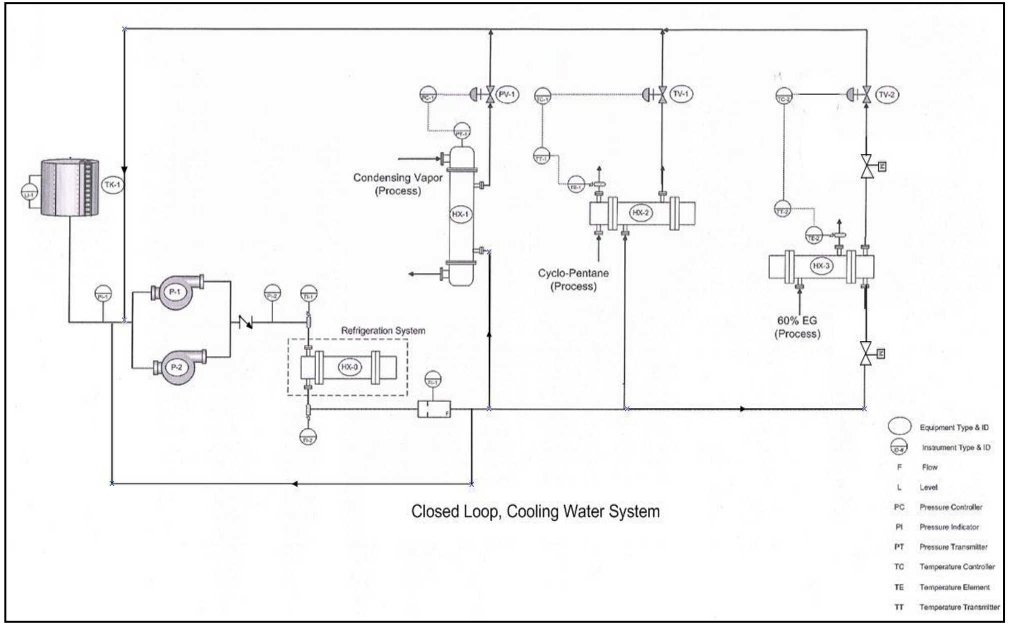
System Performance and Components Specification

NIST CRE

10/4/2012

A description of the CHL system composition and high/low level requirements in terms of performance and interfaces. This document should aid in further defining the CHL system for adapting to other interfacing systems.

Closed loop, Heat transfer, Liquid circulating (CHL)



May include:

- System Requirements Tree
- System Requirements
- Component Requirements
- Interface Requirements

How to define a language for a given domain: steps

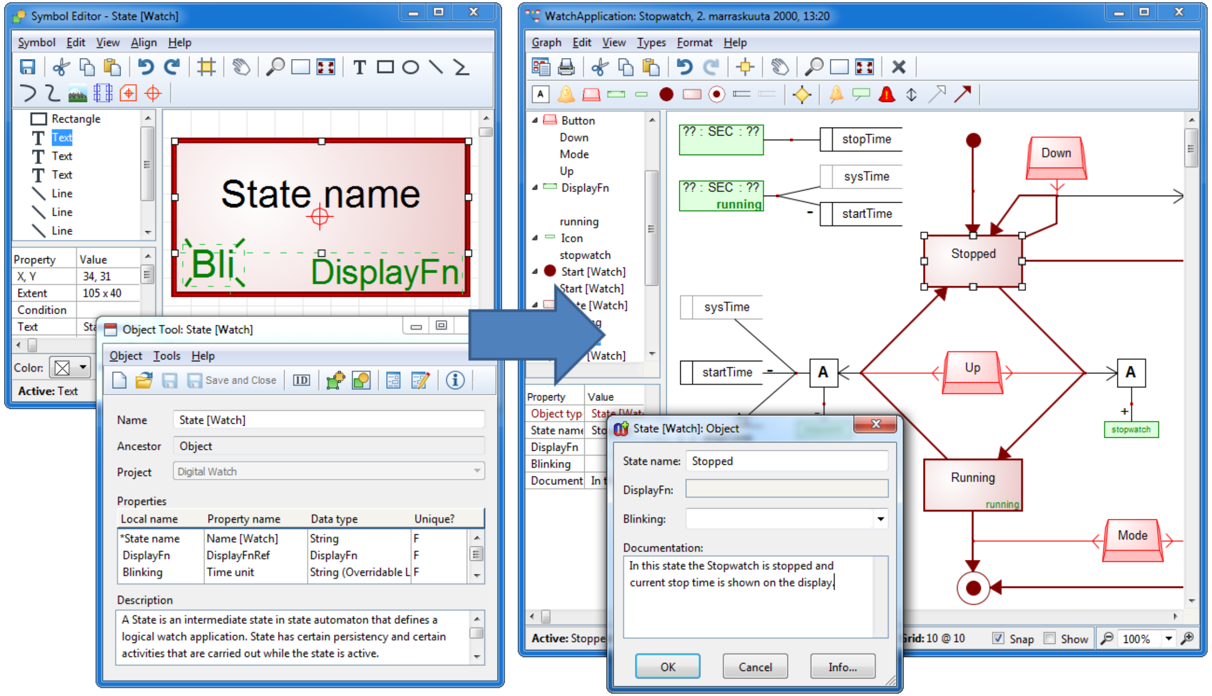
1. Identify abstractions
   - Concepts and how they work together
2. Specify the metamodel
   - Language concepts and their rules
3. Create the notation
   - Representation of models
4. Define the generators
   - Various outputs and analysis of the models

- The process is iterative: try solution with examples
  - Define part of language, model with it, define more...

Roles for language definition & use

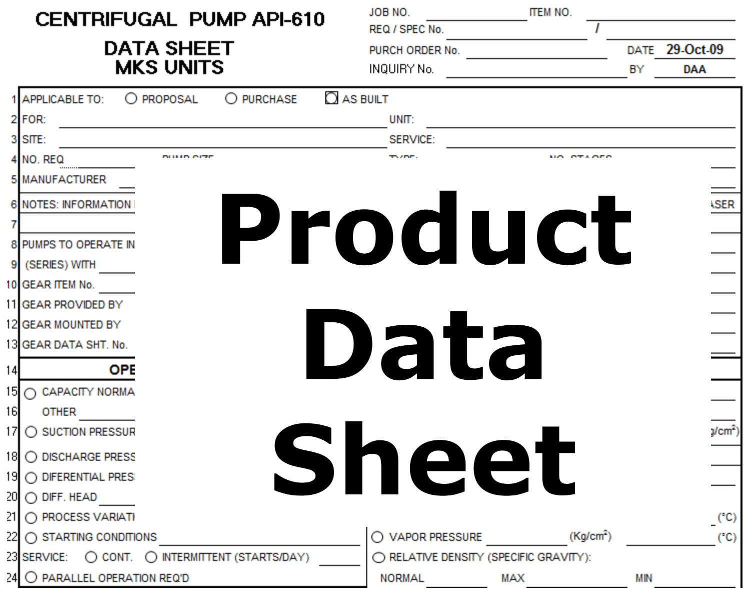
Experts define languages & generators

Team models with domain concepts & generate code, tests…



Step 1) Identify abstractions from domain terminology

- Detailed information specifying functional & physical characteristics of a component of a system, plant or facility (e.g. pump)



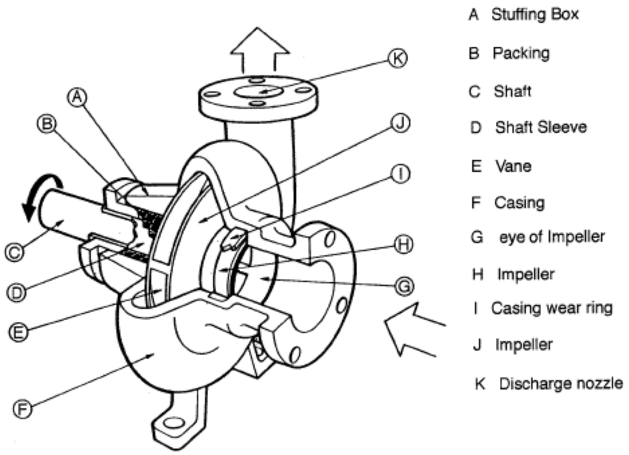
CENTRIFUGAL PUMP API-610

DATA SHEET
MKS UNITS

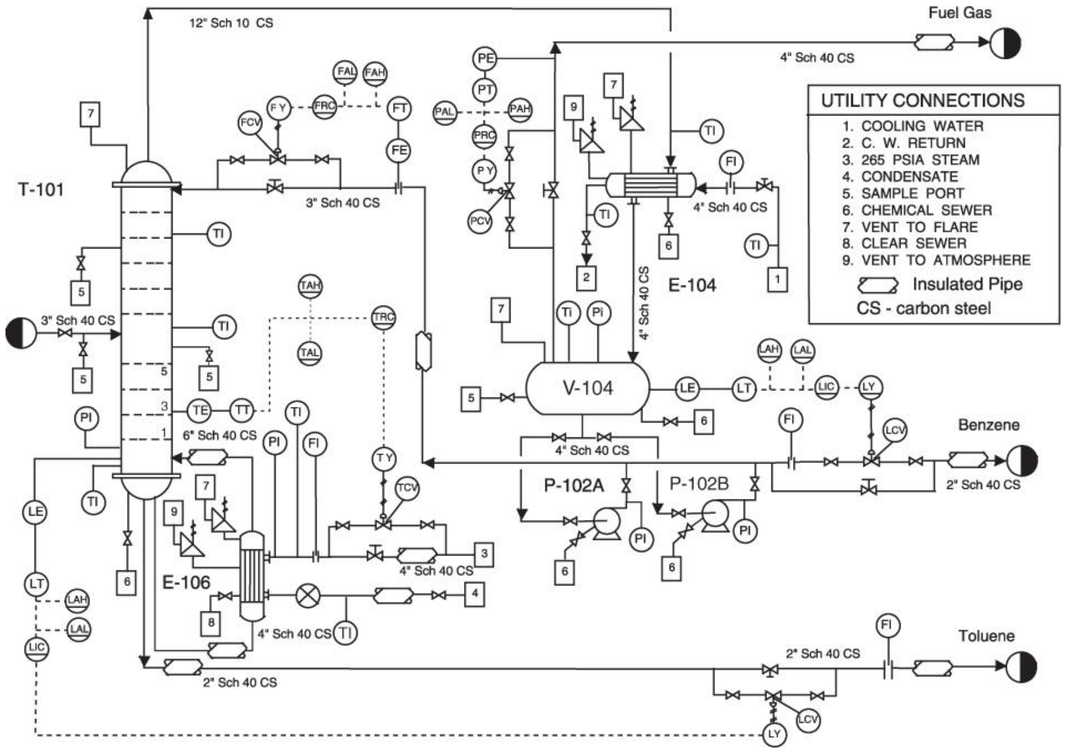
JOB NO. ______ ITEM NO. ______
REQ / SPEC No. ______ / ______
PURCH ORDER No. ______ DATE 29-Oct-09
INQUIRY No. ______ BY DAA

1 APPLICABLE TO: ○ PROPOSAL ○ PURCHASE ☐ AS BUILT
2 FOR: ______ UNIT: ______
3 SITE: ______ SERVICE: ______
4 NO. REQ
5 MANUFACTURER
6 NOTES: INFORMATION
7
8 PUMPS TO OPERATE IN
9 (SERIES) WITH
10 GEAR ITEM No.
11 GEAR PROVIDED BY
12 GEAR MOUNTED BY
13 GEAR DATA SHT. No.
14 OPE
15 ○ CAPACITY NORMA
16 OTHER
17 ○ SUCTION PRESSUR
18 ○ DISCHARGE PRESS
19 ○ DIFERENTIAL PRES
20 ○ DIFF. HEAD
21 ○ PROCESS VARIATI
22 ○ STARTING CONDITIONS ______ ○ VAPOR PRESSURE ______ (Kg/cm²) ______ (°C)
23 SERVICE: ○ CONT. ○ INTERMITTENT (STARTS/DAY) ______ ○ RELATIVE DENSITY (SPECIFIC GRAVITY):
24 ○ PARALLEL OPERATION REQ'D NORMAL ______ MAX ______ MIN ______

Product Data Sheet

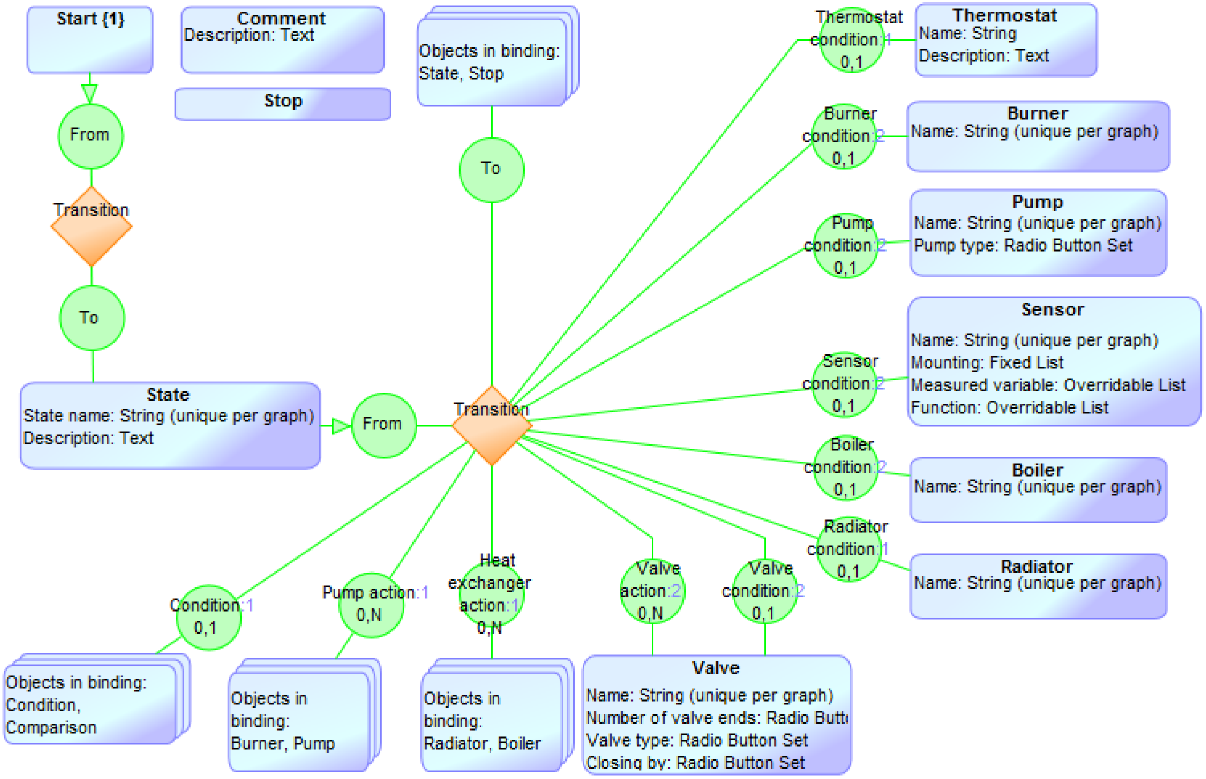


Design with domain-concepts & abstractions

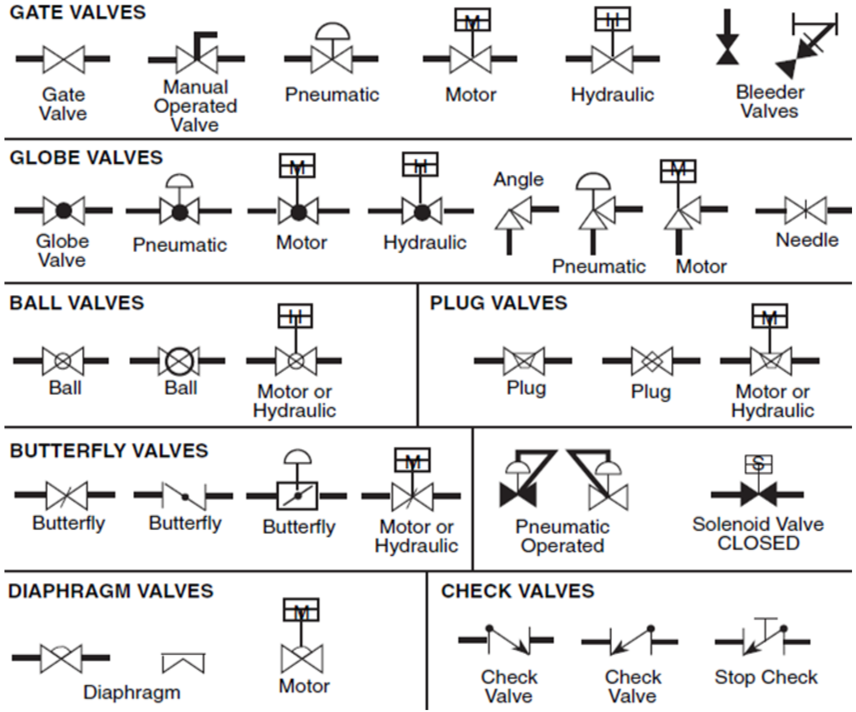


* Turton et al., Analysis, Synthesis and Design of Chemical Processes, Prentice Hall. 2012

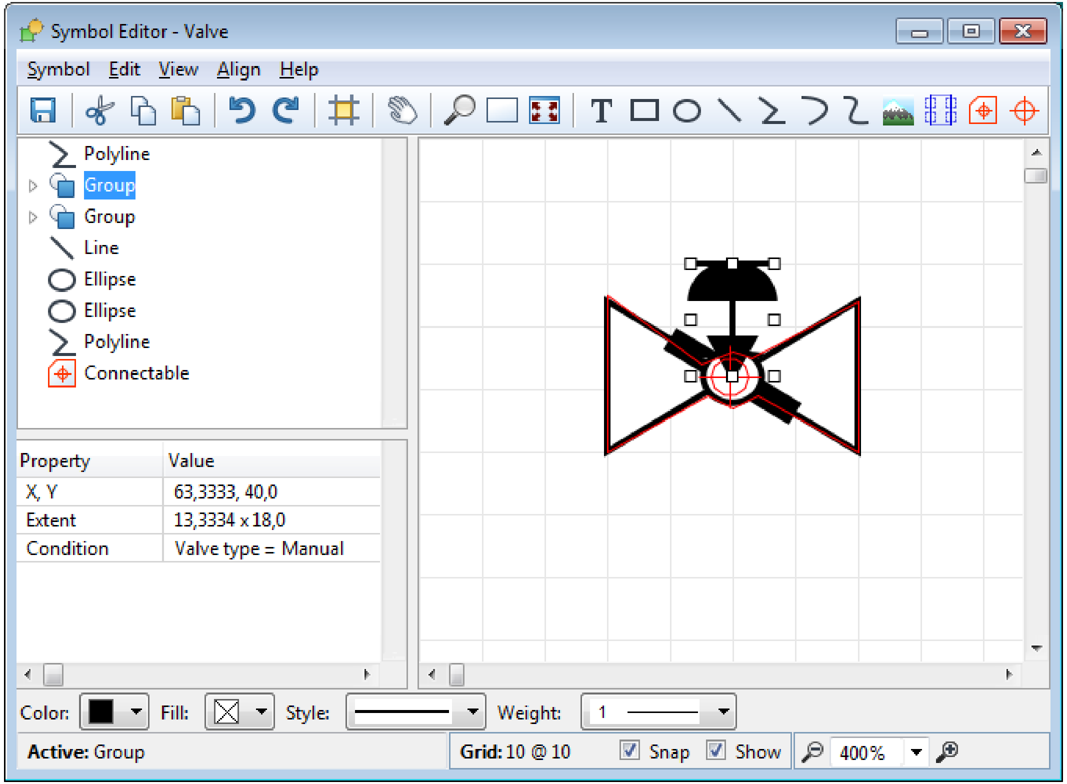
Step 2) Specify the metamodel



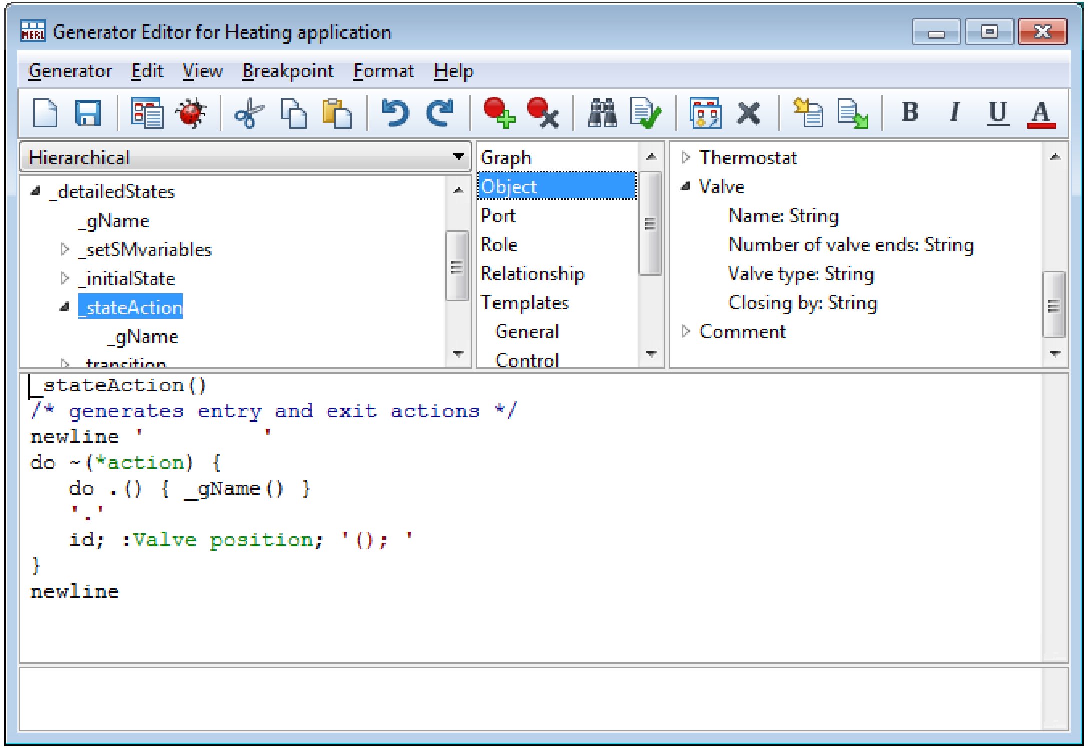
Domain terminology and visualization: Valves



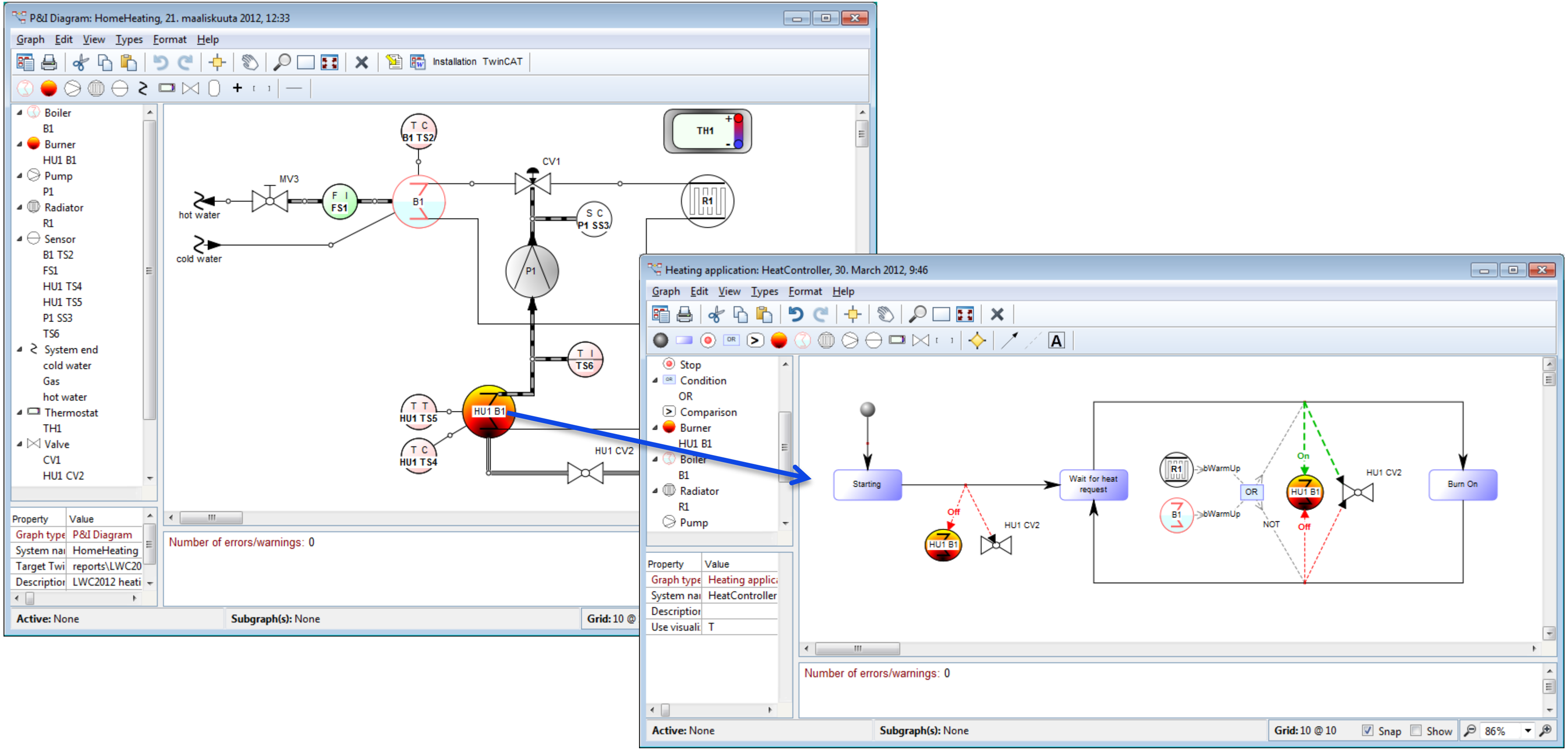
Step 3) Create the notation: Example on Valve



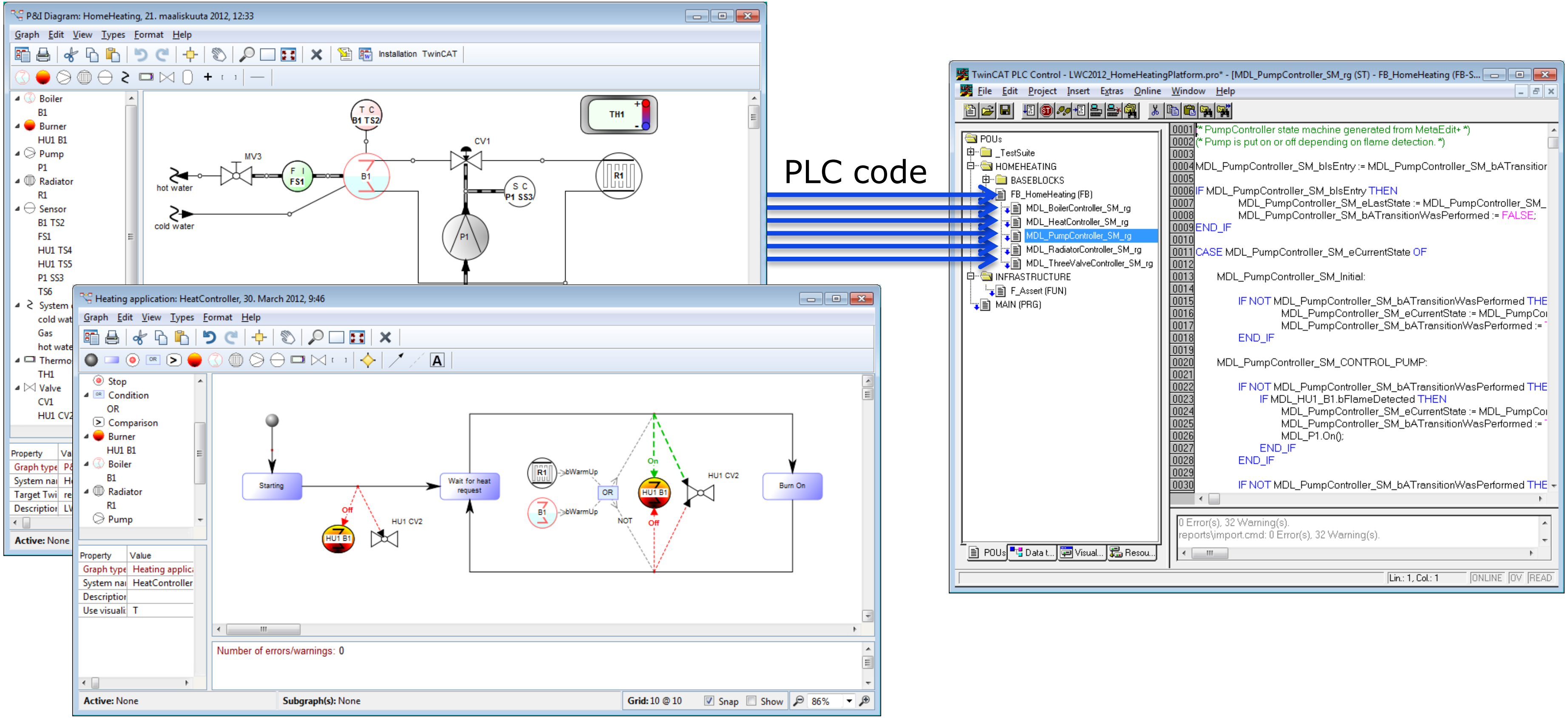
Step 4) Define the generators: PLC code for Valve actions



Developing the system with DSM



Generating the code from models



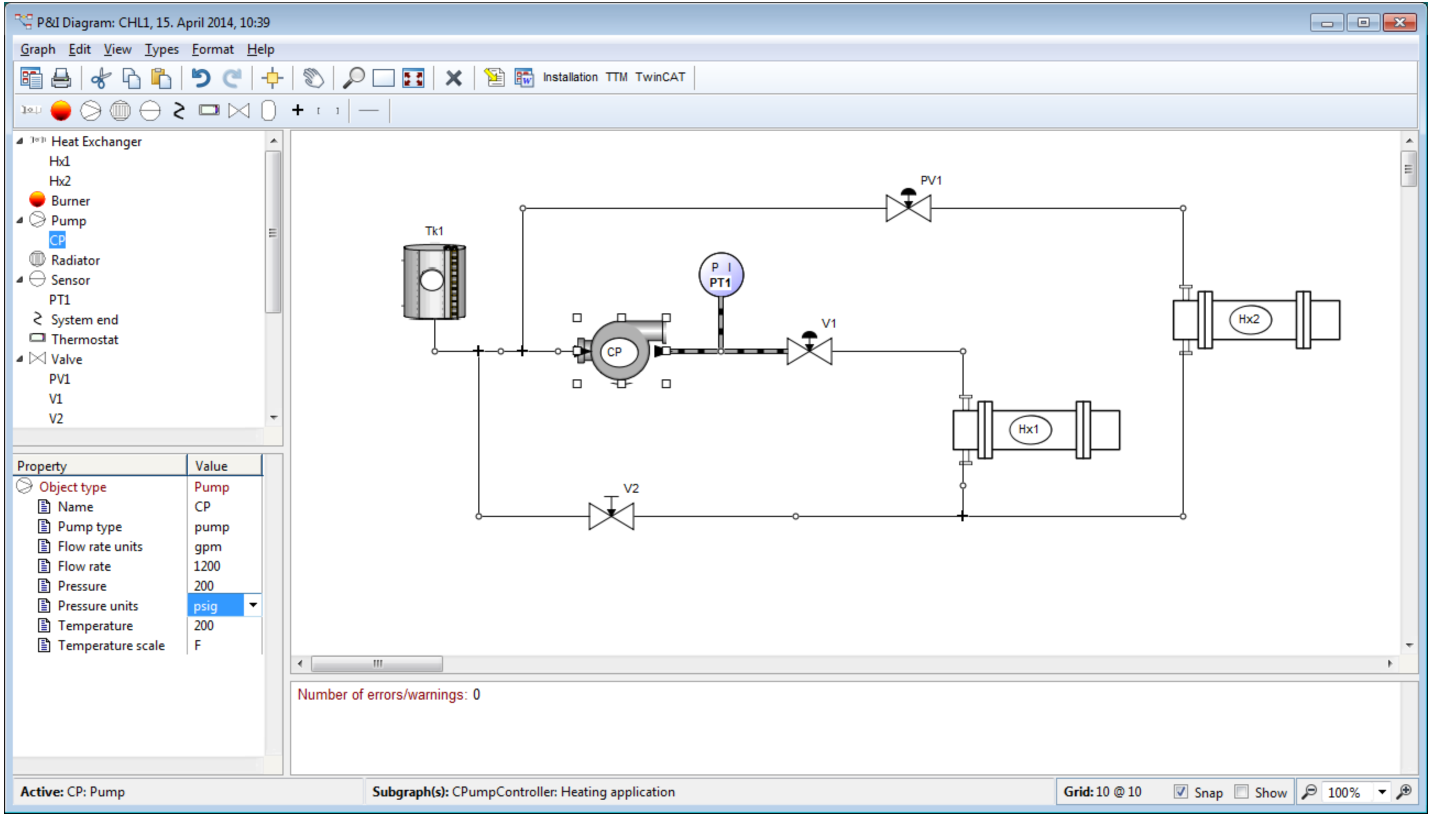
How to test a cooling system?

1. Design time
   - Domain-Specific Modeling Language already captures several rules of the system
   - Language prevents errors already in the design stage
2. Testing and V&V
   - DSL can capture aspects related to testing and V&V
   - Same language concepts used for both development as well as for testing

How to test a cooling system?

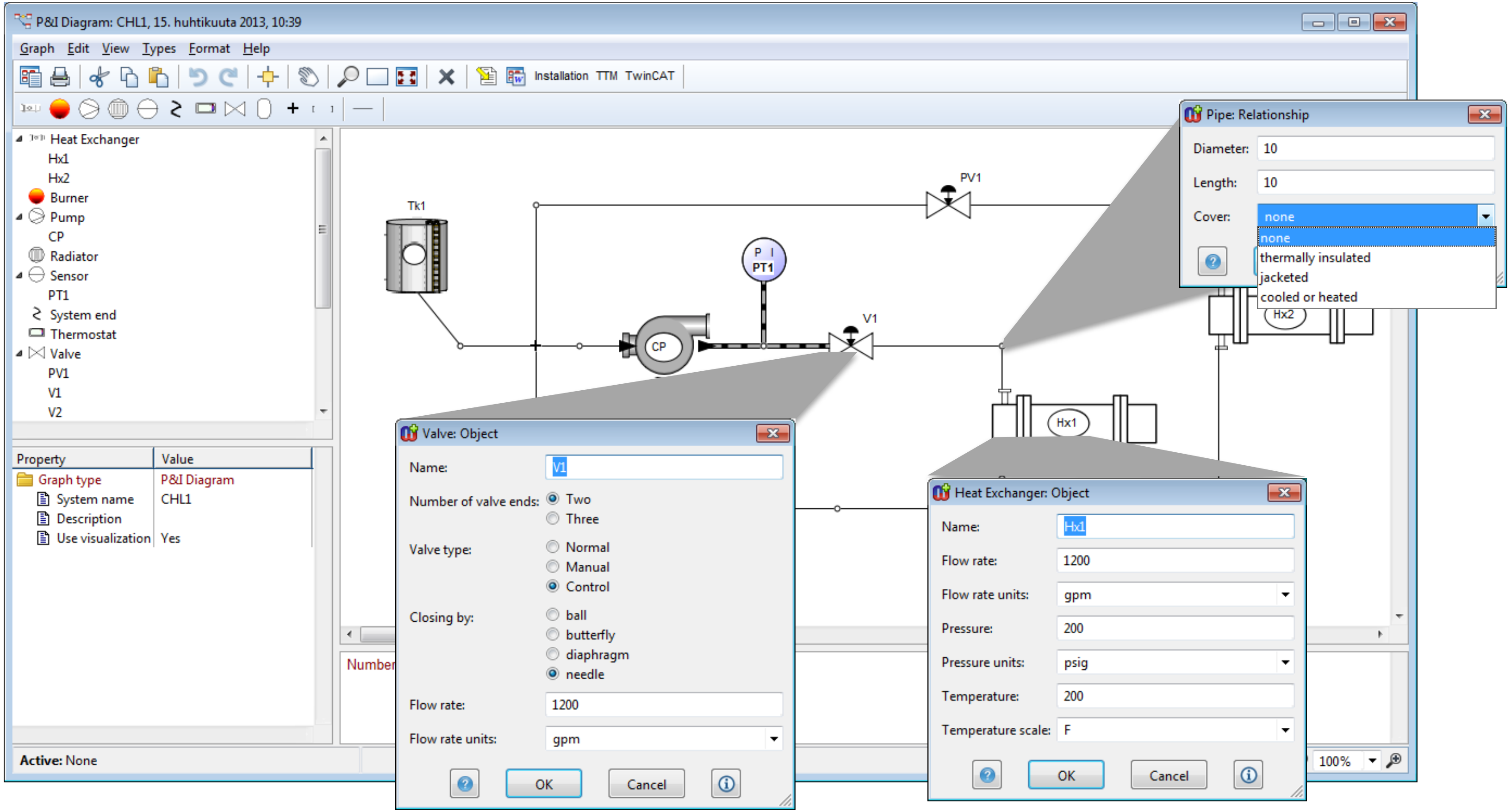
- As components (e.g., pumps, values, heat exchanger) wear out, new components are substituted
  - Common for original requirements or design to not exist
  - May not know how current facility implementation deviates from original design or requirements
- Concern: newly substituted component can create potential operational or safety issues such as:
  - Temperature: Produce too much heat?
  - Pressure: Incorrect input/output pressure?
  - Flow rates: Conflicting flow rates in the configuration?
  - Control logic errors…
  - Instrument configuration…

Example: Cooling in process plant*

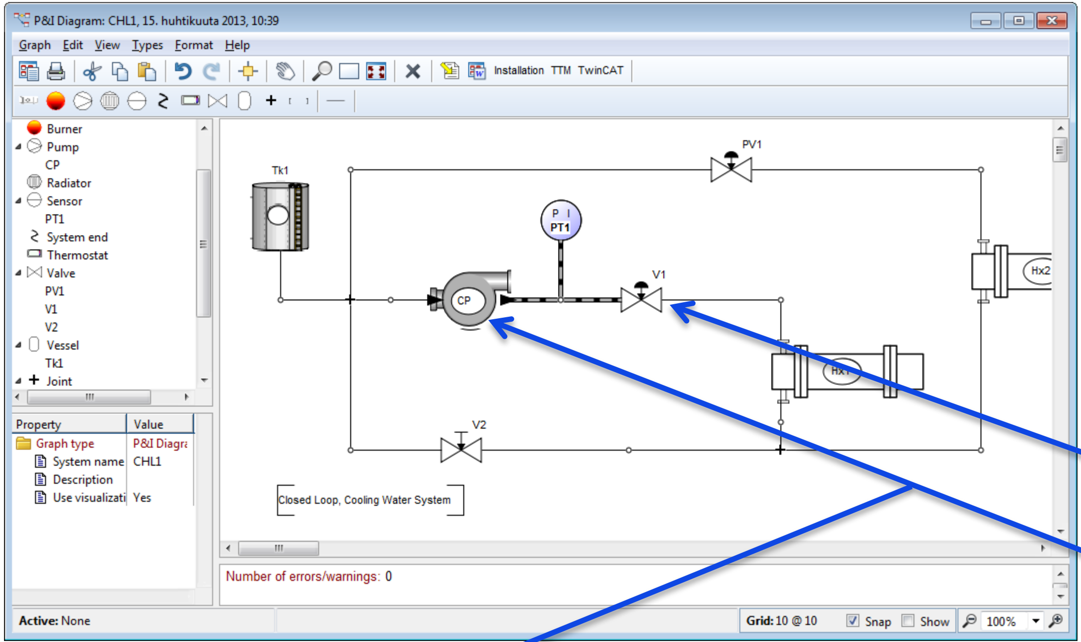


* M. Blackburn, P. Denno, Virtual Design and Verification of Cyber-Physical Systems

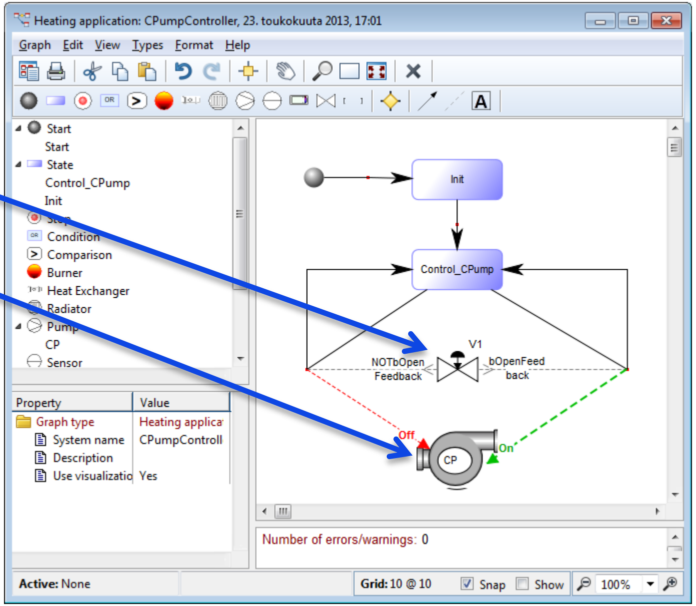
Specifying properties of components



Both structure and behavior



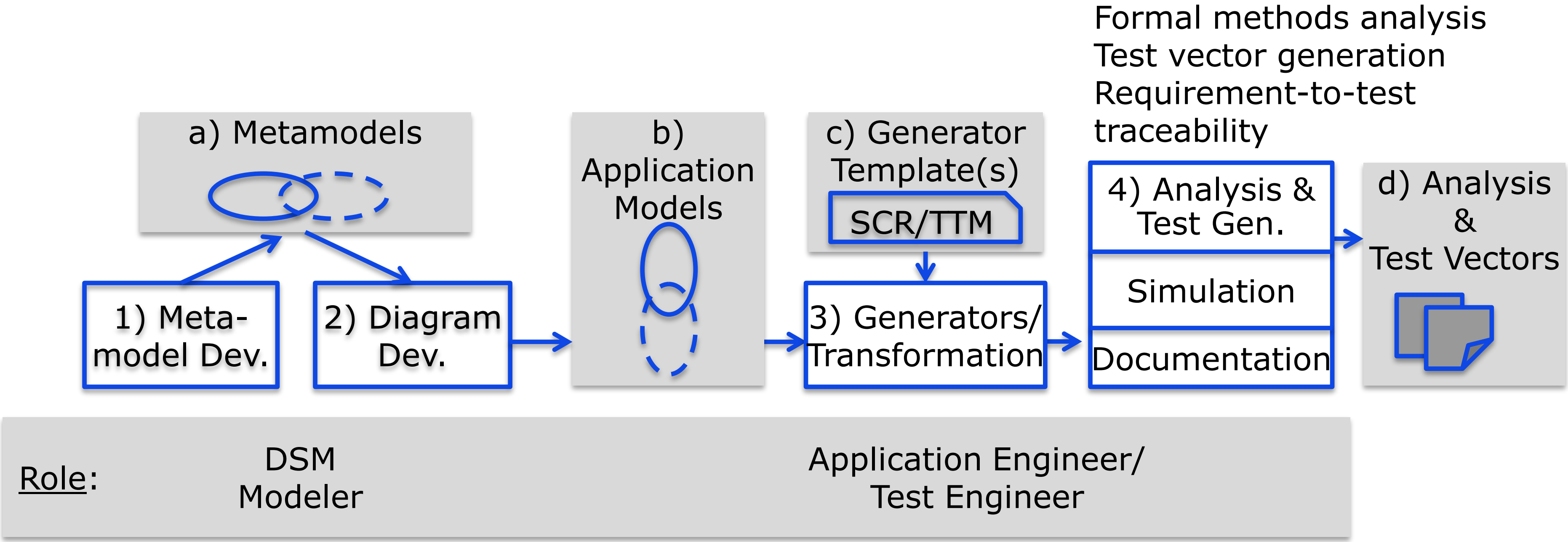
Behavioral constraint:
if valve is closed then
pump should be closed
else if value is open then
pump can be open



- Same objects: different views used to formalize different aspects of the system
- Languages integrated: can share objects used in different diagram types

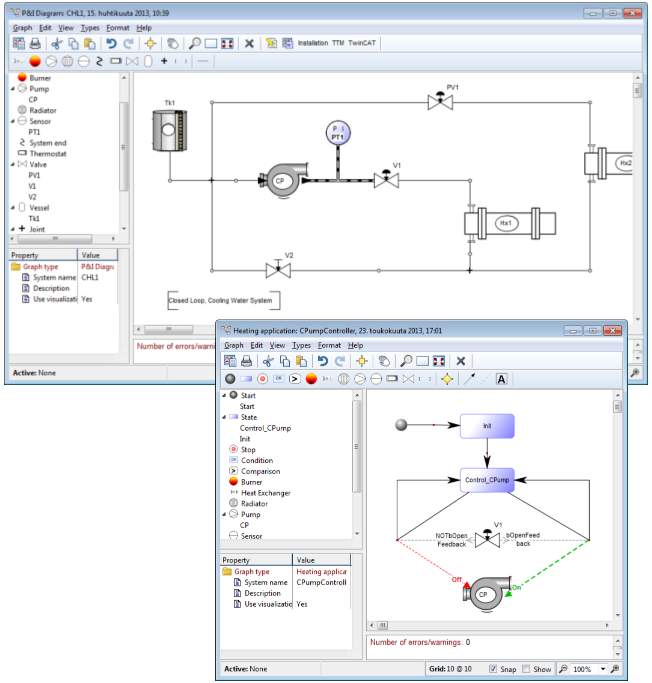
Toolchain and stakeholder roles

- Conceptual representation of DSM and formal methods toolchain

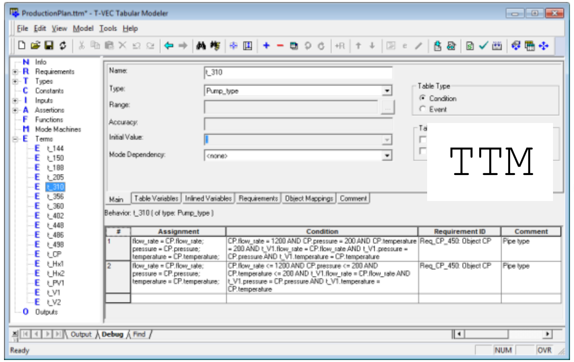


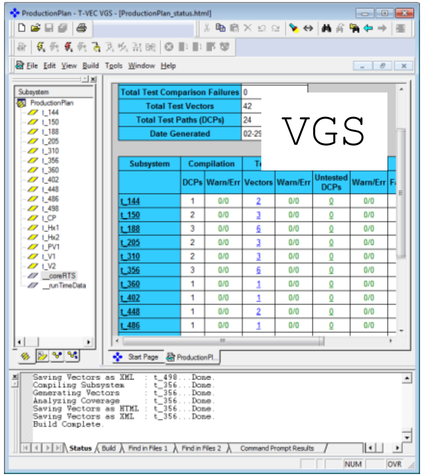
DSM integrates with formal methods

MetaEdit+ -based DSM
for Process Facility Design



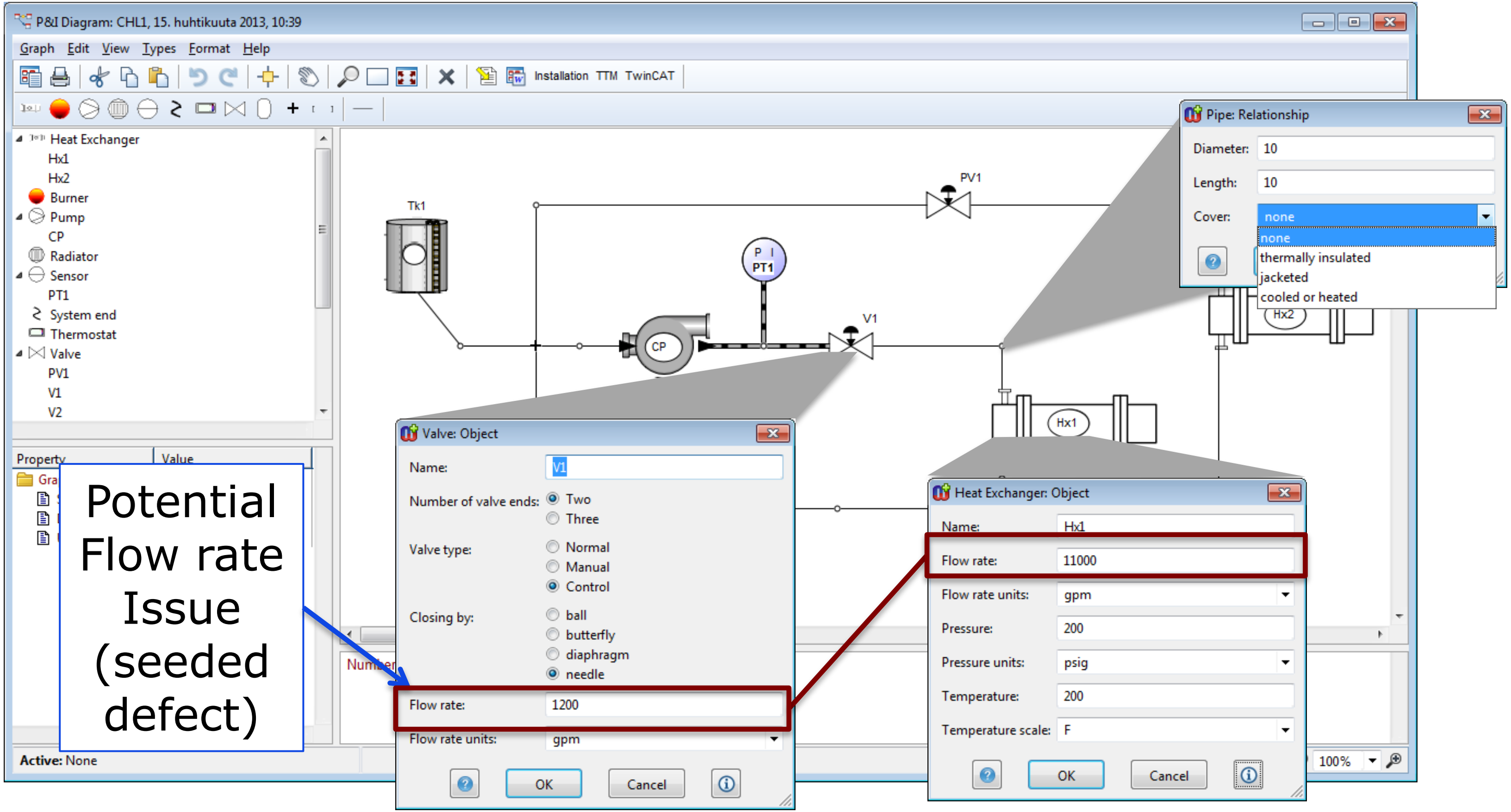
T-VEC Tabular Modeler (TTM) and
T-VEC Vector Generation System (VGS)





- Model transformation
- Formal methods analysis
    - Theorem proving
    - Property checking
- Test vector generation
- Test driver generation
- Requirement-to-test traceability

Model captures component properties



Analysis identifies unsatisfiable constraints

t_150 Coverage Analysis FAILED

Vector File: C:\Users\jpt\Documents\MetaEdit+ 5.1\reports\test_vectors\t_150.TST

Time Run	02-29-16 17:37:27	
Analyzer Version	4.0.0	
Total Number of DCPs	2	
DCPs Not Covered	1	
Predicates Requiring Coverage	11	
Predicates Not Covered	5	cv_t_150_RP__1 t_150 t_150_FR__1 t_150_RP__1 t_150_1__LS
Total Coverage Errors	6	
Total Coverage Warnings	0	
Test Generation Failures	2	

TTM Table t_150

#	Assignment	Condition	Requirement ID

1	flow_rate = Hx1.flow_rate;
	pressure = Hx1.pressure;
	temperature = Hx1.temperature;
		Hx1.flow_rate = 11000 AND Hx1.pressure = 200 AND Hx1.temperature = 200 AND t_V1.flow_rate = Hx1.flow_rate AND t_V1.pressure = Hx1.pressure AND t_V1.temperature = Hx1.temperature	Req_Hx1_468: Objec...

2	flow_rate = Hx1.flow_rate;
	pressure = Hx1.pressure;
	temperature = Hx1.temperature;
		Hx1.flow_rate <= 11000 AND Hx1.pressure <= 200 AND Hx1.temperature <= 200 AND t_V1.flow_rate = Hx1.flow_rate AND t_V1.pressure = Hx1.pressure AND t_V1.temperature = Hx1.temperature	Req_Hx1_468: Objec...

Hyperlink from analysis report to model defect

Uncovered DCP Paths

DCP Number	DCP Path	Failure Detection
1	t_150, t_150_FR__1, cv_t_150_RP__1, t_150_RP__1, t_150_RP__0, t_150_1__LS (goto model)	Vector Generator

Failed Pre-Condition Relation	cv_equal_to	
Exact SS File Location	t_150.SS	Line # 30
Input Domain At Error	Occurance at UA = 662 t_V1__VAR.flow_rate:Valve_type_flow_rateDataType [0 .. 1200] Hx1.flow_rate:Heat_Exchanger_type_flow_rateDataType [11000 .. 11000]	

Experiences from the CHL case

- Model-based approach to:
  - Analyze requirements and create design specifications
  - Generate implementation code, deployment, docs etc.
  - Generate tests and provide traceability of tests to corresponding requirements
- Benefits
  - Improved system validation
  - Ability to better trace rationale
  - Improved systems engineering

Why DSM: emerging and enabling technology for V&V

- Provides relevant and intuitive graphical abstraction for specific domain or related subdomains
- Allows for rich semantics required for formal analysis and test generation
  - Necessary for V&V effectiveness and efficiency
- DSM tooling allows multiple views to be integrated
  - Model transformation often built into the tools
  - Integrates formal analysis and test generation tools
  - Formal methods hidden behind the scenes
  - Model languages are evolvable

Summary

- DSM provides differing views of system designs across multiple disciplines
  - Approach enables collaboration between domain experts and developers, development and verification
  - Leverage and integrate new analysis and model-based automation (e.g., simulation, synthesis, generation, etc.)
- Future challenges – increase in modeling & analysis tools
  - Apply further views and multi-model consistency (e.g. in automotive)
  - Integrate with various testing tools, each focusing on different aspects

MetaCase

Thank you!
Questions, please?

For references on examples and cases contact:

Juha-Pekka Tolvanen, jpt@metacase.com

www.metacase.com

References

- Blackburn, M., Denno, P., Virtual Design and Verification of Cyber-Physical Systems: Industrial Process Plant Design, Procedia Computer Science 28, Elsevier, 2014
- Kelly, S., Tolvanen, J.-P., Domain-Specific Modeling: Enabling Full Code Generation, Wiley, 2008. http://dsmbook.com
- EADS, www.metacase.com/papers/MetaEdit_in_EADS.pdf
- Elektrobit, O.-P. Puolitaival et al, Utilizing Domain-Specific Modeling for Software Testing, Proceedings of VALID, October 2011
- NSN, Architecture in the language, www.metacase.com/cases/architectureDSMatNSN.html
- Nokia, www.metacase.com/papers/MetaEdit_in_Nokia.pdf
- Panasonic, Proceedings of Domain-Specific Modeling, 2007, www.dsmforum.org/events/DSM07/papers/safa.pdf
- Polar, Proceedings of Domain-Specific Modeling , 2009, www.dsmforum.org/events/DSM09/Papers/Karna.pdf
- USAF, ICSE, http://dl.acm.org/citation.cfm?id=227842